תרגיל בית 5

234124 - מבוא לתכנות מערכות

שם: פרח קב

ת״ז: 213309388

שם: ג׳מאל אבו מוך

ת״ז: 213754088

חלק יבש 3

3.1 the ord() function in python is used generally to get the Unicode code (the ASCII value) of a given character, returns an integer that represents the Unicode code from the Unicode standard.

```
my_list = []
index = 0
while index < len(my_str):

    my_list.append(ord(my_str[index]))

    index += 1
```

this main idea of this code is to save the Unicode code for each character in the given string ’my_str’ in a list ‘ my_list’. At first, initializing an empty list ‘my_list’ , then setting an int index with a starting value 0, then there is a while function that checks every iteration: if the index is equal to the length of the given string with the function len(seq) that return the length of a sequence, in the while loop, appends the Unicode code of the current character in the string ‘my_str’ to the list ‘my_list’ using the ‘ord()’ function and adding the index with 1 (number of rounds).

We can replace the given code with a single line with the same result using a list comprehension:

```
my_list = [ord(c) for c in my_str]
```

3.2 in this code

```
my_dictionary = {}


for x in [1,2,3,4,5,6,7,8,9,10][::-1]:

      my_dictionary[x] = f"{x}^{x} is {x ** x}\n"

print(*my_dictionary.values())
```

creating an empty dictionary, then iterates over a list of numbers from 10 to 1, because of [::-1] with no start or end and only with step_size = -1, this creates a reversed version of the list,for each number x in the list , it puts the number in the dictionary as the key, and the value of this given key is the value of this string `f"{x}^{x} is {x ** x}\n"`,
that shows the number ‘x’ ,the exponentiation operation ‘x^x’ and the result of that operation ‘x**x’ ,after that it prints all the values in the dictionary, the ‘`my_dictionary.values()`’
produces a view object that contains the values like retrieves the values from the dictionary, and the ‘*’ operator unpacks these values so that each value is treated as a separate argument to ‘print()’, separated by spaces.
and the result will be the values of the dictionary in a single line, separated by spaces, the values are from 10 to 1 ‘10^10 is 10000000000’ …
We can replace the given code with a single line with the same result by creating and printing the formatted strings directly:

print(*[f”{x} ^ {x} is {x ** x}” for x in range(10, 0, -1)])

3.3 the ‘ord()’ function is often used in conjunction with the ‘chr()’ function, which does the reverse operation by converting a Unicode code point back into character.

```
for i in range(ord('A'), ord('z')+1, 2):
    if chr(i).isalpha():

print(f"The ASCII number {i} represent the char {chr(i)}")
```

the code loops through every other ASCII value between ‘A’(65) and ‘z’(122), checks if the character corresponding to that ASCII value is an alphabetic letter, and then prints the ASCII value and the character if it’s indeed a letter.
The ‘range()’ function in python returns an iterable object which represents the requested range of members from a type int, the range() function is exclusive of its stop value, so to include the character ‘z’ we add 1.In the given code `range(ord('A'), ord('z')+1, 2):`
This creates a range of numbers starting from 65(ASCII of ‘A’) to 123 (one more than the ASCII of ‘z’), with a step of 2. The step of 2 means it will skip every even number.
The isalpha() function checks whether the character is an alphabetic letter (A-Z or a-z). it returns ‘True’ if the character is a letter and ‘False’ otherwise. Then it prints a message that shows the ASCII number of the not even numbers between 65 and 123 with the corresponding character.
We can replace the given code with a single line with

[print(f"The ASCII number {i} represent the char {chr(i)}") for i in range(ord(‘A’), ord(‘z’) + 1, 2) if chr(i).isalpha()]

3.4

```
tmp_chr = ""
for num in list_c:
    tmp_chr += chr(num)
print(tmp_chr)
```

this code takes a list of integers 'list_c' and coverts each integer into its corresponding ASCII character, and then combines all these characters into a single string, then it prints this string.

The output for the given `list_c = [80, 121, 116, 104, 111, 110, 32, 105, 115, 32, 102, 117, 110, 33]`

```
Is Python is fun!
```

We can use the 'join' method that takes all the characters generated by the generator expression and joins them into a single string, with no separator between them:
The online code would be

print("".join(chr(num) for num in list_c))